محاضرة 6 : أساسيات الـ intensity mapping

المحتوى :
① تصنيف ال Spatial processing
② أساسيات ال Spatial processing
③ دوال ال intensity mapping
Dynamic Range stretch and threshold
Negative
Logarithmic
Power Law trans.
④ Piecewise Linear Transformation
⑤ Intensity Level and bit plane slicing

تصنيف ال Spatial processing

* ممكن نقسمها لحاجتين
① intensity Transformation : بيشتغل فيها على بكسل واحد بس ، وممكن نصيغ الكلام بأننا شغالين على neighborhood size بيساوي 1x1
- من التطبيقات عليه هي ال Contrast Manipulation and image thresholding
* ممكن نعتبره حالة خاصة من ال spacial filtering
② Spatial filtering : بيشتغل على ال neighborhood rectangle ، يعني عندي window ليها أبعاد MxN زي ما شغلنا في ال blur في محاضرة 4
- من التطبيقات عليه هي ال Contrast Manipulation and preprocessing
* لو خليت ال window حجمها 1x1 يبقى شغال Intensity transformation

---

أساسيات ال Spatial processing

* عمليات ال Spatial processing عادة بتكون Effecient وبتستهلك Resources أقل بالمقارنة بال Transform operations

* الصيغة العامة هي

$$g(x,y) = T[f(x,y)]$$

- حيث f(x,y) هي ال input image
- ال g(x,y) هي ال output img

- ال T[ ] هي ال operation اللي هعملها سواء على بكسل واحد أو ال neighbors

٧ هنكمل نص اللي قولناه في كاميرة 5 صفحة 3 لو كان عندي pixels خارج حدود الصورة بسبب الـ window عندي حل من اتنين:

- اما اني أفرض قيم للـ pixels (Padding out of image pixels with a specific value)

- أو اني أتجاهل قيمة الـ pixels دي نهائي

مثال على العملية T هي عملية الـ averaging (قولناها قبل كده) اللي بتسبب blur

* عملية الـ T Transformation مع الـ Neighborhood Rectangle (اللي هي الـ window بتاعتي) بنسميها الـ Spatial filtering ممكن نسمي الـ neighborhood Rectangle بأسماء تانية زي

- Spatial Mask - Kernel - Template - Mask

* هنكر تاني إن الـ intensity mapping حالة خاصة من الـ spatial transformation لما يكون الـ rectangle أبعاده 1 x 1 (بشتغل على البكسل نفسه ومليش دعوة بالـ neighbors)

- في الحالة دي بتبقى المعادلة

$$S = T(r)$$

حيث r هي قيمة الـ intensity في الـ input

S هي قيمة الـ pixel في الـ output (نفس الـ location)

و T هو الـ operator أو العملية اللي بعملها

* هنكر تاني: من التطبيقات على الكلام ده هو الـ image Enhancement

---

دوال الـ intensity mapping

* الهدف الأساسي منها هو الـ image enhancement

* الـ image enhancement مش مقتصر على تحسين الصورة لل human vision وبس ممكن أكون عايز أحسن الصورة عشان تدي نتايج أحسن في حاجة تخص ال computing زي إني أحسن الصورة بحيث تدي recognition rate أعلى في تطبيق زي ال OCR

- في الحالة دي بيكون المعيار عندي مش شكل الصورة اللي بشوفه لكن ال recognition rate

* يبقى ممكن نصنف جودة وطريقة ال Enhancement اعتمادًا على حاجة من اتنين:

1. في حالة تحسين الصورة لل human investigation بيكون اختيار طريقة ال Enhancement خاضع لل subjective judgment للي هيعمل ال investigation

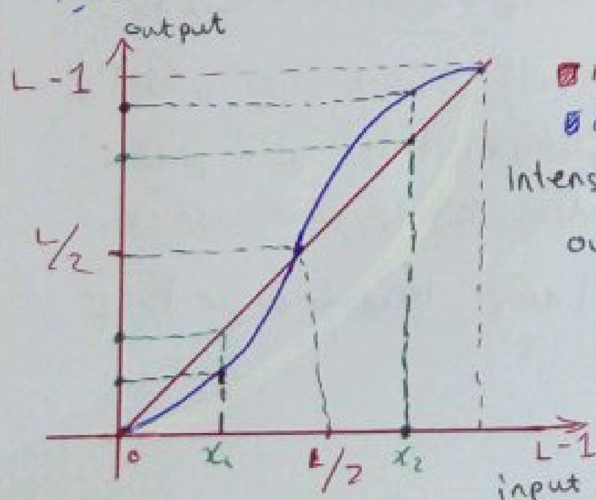
2. في حالة تحسين الصورة لل Automated machine processing بيبقى المعيار عندي objective زي ال recognition rate في ال OCR

---

* الدوال مشروحة كويس في ورق السنة اللي فاتت صفحة 25، 26، 27، 28 بترقيم البرنامج مش الصفحات

* هنوضح الفكرة ورا الرسم بتاع المعادلات عشان تفهم إزاي المعادلات بتعدل في ال intensity وبعدها شوية ملاحظات

* ابقى شوف ال Slides باعرة مع الورق ده من أول سلايد [6.6] لحد سلايد [6.17] هتلاقي شوية تطبيقات معاها وصور توضح أثر كل عملية



* لو فرضت محور أفقي عليه ال intensity بتاعت ال input image والمحور الرأسي عليه ال intensity بتاعت ال output، في حالة إني مش هغير ال Intensity هتبقى العلاقة بين ال output وال input إنه output = input وتدي علاقة Linear بالأحمر

- لو جبت أعلى البكسلز ناحية الأبيض (L-1) أفتح وجبت أغمق ال pixels أقرب للأسود (0) هعمل ال Mapping بالأزرق

* بنفس المنطق هيمشي بيك على كل ال functions

- ال Thresholding لو قيم البكسل أقل من K هتبقى أسود، ولو أكبر منك هتبقى أبيض ، ده هيحول الصورة ل Binary image

* تطبيقات كل function موجودة في المحاضرات وورق السنة اللي فاتت ، ركزوا فيها كويس.

* في ال negative function هنعكس قيم ال pixels ، اللي كان أبيض تخليه أسود واللي كان أسود تخليه أبيض.

* ركز على معرفة تطبيق كل function ، ركز على معرفة تطبيق كل function

---

Piecewise linear transformation

المفروض تكون ذاكرته من الورق وال slides لو خلصت زي ما قولتلك في اللي فات

Intensity Level Slicing & bit-plane slicing

جزء ال Intensity Level Slicing هيشرح كويس في السلايد 6.17 ، والصور بتوضح التطبيق على كل function

الصورة (A) هي الصورة الأصلية ، الصورة (B) هي تطبيق على ال function اللي هي (a) ، والصورة (C) تطبيق على الدالة (b)

* نشرح ال bit-plane slicing بشوية تفصيل عشان مش مترجمة كويس في ال slides

* ترقيم ال planes وال reconstruction للصورة فيه غلطة في ورق السنة اللي فاتت ، خلي بالك . خدها من هنا والسلايد رقم [6.20] ، [6.21]

* الفكرة: هنا إن عندي الصورة كل بكسل فيها قيمته بتتخزن في مجموعة من ال Bits ، ممكن آخد أول Bit لكل ال pixels وأعمل منه صورة Binary ، وتاني Bit آخده لكل ال pixels وأعمل منه صورة Binary وهكذا

* كل صورة بنسميها plane ، لو عندي K=8 يبقى ممكن أعمل 8 planes

* ترقيم ال planes بيبدأ بـ[1] وبينتهي بـ k ، بينما ترقيم ال Bits بيبدأ من صفر وبينتهي بـ (k-1)

يعني

plane 1 → hold bit-0
plane 2 → hold bit-1
⋮
plane k → hold bit-(k-1)

* ال bits الصغيرة زي bit-0, bit-1, bit-2 تأثيرها في الصور مش كبير
* ال bits الكبيرة زي bit-5, bit-6, bit-7 بيبقى تأثيرها كبير ، وممكن أكتفي في تطبيق زي ضغط الصورة (Compression) إني أحفظ قيم ال bits اللي هي (7, 6, 5) بس وأهمل الباقي (الصورة هتقل جودتها بس العين مش هتلاحظ الفرق أوي)

* عشان تشوف الصورة شكلها عامل إزاي في كل plane شوف سلايد [6.19]

* السبب إن ال bits الكبيرة أثرها أهم إنها بتدي قيم أكبر

7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0,128	0,64	0,32	0,16	0,8	0,4	0,2	0,1

هتلاقي ال bit رقم 7 مسؤولة إنها تدي 128 أو صفر ، والرقم 128 لما نيضاف للصورة يدي تأثيراً كبير بكتير من مثلاً ال bit-1 اللي بتدي [0, 2]

* التطبيق في سلايد 6.20 على ال Compression

* إزاي أعمل ال bit-plane slicing ؟
فيه طريقة حلوة (بس الدكتور [مشطوب] مش مقتنع بيها أوي وعايز الطريقة اللي بيرحها) عن طريق ال Masking زي ما كنا بنعمل في ال micro
* لو عندي قيمة بكسل 11010111 = 215
ممكن أعمل AND مع كل bit وآخدها لوحدها.

الطريقة اللي عايزها الدكتور مختلفة شوية، هنقسم قيم ال pixels لـ Ranges
وحسب ال Range اللي فيه ال pixel بقول هو صفر ولا 1 في ال plane

b_2	b_1	b_0	Value
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

هناخد مثال على صورة فيها K=3 للتبسيط

plane 1 - bit 0
if value is odd then bit = 1
else then bit = 0

plane 2 - bit 1
if value $\in [0, 1]$ ← تنتمي للـ Range
or value $\in [4, 5]$
then Bit = 0
if value $\in [2, 3]$
or value $\in [6, 7]$
then Bit = 1

plane 3 - bit 2
if value $\in [0, 3]$ then bit = 0
if value $\in [4, 7]$ then bit = 1
OR
if value ≥ 4 then bit = 1
else then bit = 0

* إزاي أرجع الصورة؟
لضرب كل bit في ال base بتاعها وأجمعهم، يعني لو فرضنا عندي
صورة محتفظ بال planes رقم 6, 7, 8 وفيه pixel قيمتك (ال bits يعني
في كل plane) p@plan 8=1 , p@plane 7=1, p@plane 6=0
هتبقى قيمة البكسل لما أرجعه

$$P_{value} = \underset{\text{Plane 8}}{1 * 2^7} + \underset{\text{plane 7}}{1 * 2^6} + \underset{\text{plane 6}}{0 * 2^5}$$
$$= 128 + 64 + 0 = 192$$